CV and IoT-based Remote Triggered Labs: Use Case of Conservation of Mechanical Energy

K.S. Viswanadh, O. Kathalkar, P. Vinzey, N. Nilesh, S. Chaudhari, V. Choppella
International Institute of Information Technology-Hyderabad (IIIT-H), India
Emails: {savitha.viswanadh, om.kathalkar, piyusha.vinzey, nitin.nilesh}@research.iiit.ac.in,
{sachin.chaudhari, venkatesh.choppella}@iiit.ac.in

***Abstract*—Remote Triggered Labs (RTL) are helpful for students to work on laboratory experiments virtually anytime, anywhere. Such setups can facilitate distance learning and are helpful during pandemics. In this paper, the use of Computer-Vision (CV) is demonstrated for RTL experiments. For this, a use-case of the *Conservation of Mechanical Energy* experiment is considered. A CV-based approach is used to estimate an object's velocity whose setup primarily consists of a microprocessor, a camera and infrared (IR) sensors. The experiment is recorded, and various CV techniques are employed to estimate the object's velocity. This paper also compares a CV-based and a IR sensor-based approach to estimate the object's velocity. Linear regression applied on the CV-based implementation resulted in an optimal mean-squared error (MSE), nearly 10 times better than IR-based implementation.**

***Index Terms*—Object tracking; Computer Vision (CV); Conservation of Mechanical Energy; Remote Triggered Labs; Internet of Things (IoT)**

I. Introduction

Many schools and colleges in developing countries, especially in rural areas, lack access to basic laboratory facilities. During COVID-19, this situation got worse and even the educational institutes having good experimental setups could not access their labs online. Remote triggered labs (RTL), which are IoT-based labs, can come handy in such situations as the students can perform the experiments remotely on an actual experiment setup over an internet browser from anywhere in the world [1]–[3]. The user can control input parameters for each experiment, and the corresponding outputs are returned to the dashboard. These outputs can be viewed from a browser on smart devices like laptops or smartphones. The results are visualised in plots and tables to make the observations conveniently and understand the experiment easily. These labs help gain hands-on experience critical to the learning and teaching process. They help students develop scientific reasoning abilities, understand the process of scientific investigation, and develop a broad understanding of scientific concepts.

RTL can host a variety of experiments from different fields including Engineering, Physics, Chemistry and Biology. Several universities have developed experiments for remote labs that include MIT's iLabs [1], NIT Surathkal's SOLVE Lab [2] and UNILabs [3]. There has been some more work on RTL in the literature [4]–[7]. Experiments developed include determining Hooke's Law [4], determining the Young's Modulus of a specimen [5] and verification of Snell's law [6]. In [7], a reusable remote lab for teaching the subject of electronics to students is presented. However, none of them use computer vision (CV) algorithms for RTL experiments, which is the focus of this paper. CV algorithms can be used to overcome the limitations faced by measuring devices like sensors, where data collection can be limited by physical constraints like orientation, type and number of sensors that can be placed.

Specific contributions of this paper are

- A CV and IoT-based RTL implementation is proposed. The proof of concept is demonstrated for the Physics experiment of *Conservation of Mechanical Energy*.
- In this experiment, CV is used to estimate the velocity of a moving object along different points on the track, which are then shown to be close to the theoretical values found based on the principle of conservation of energy.
- To improve the performance of CV-based velocity estimation of the moving object at a given point, we use linear regression to find the line of best fit for all the points on a straight track. Also, parameters to the implementation are changed to observe their effect on estimated velocities.
- A performance comparison of CV-based implementation is carried out with infrared (IR)-based velocity estimation. It is also shown that the CV-based implementation can estimate velocity at any location on the track without the need for a sensor.

Experimental setups utilising CV-based techniques for obtaining outputs are used in [8]–[11]. In [8], a motion analysis system was developed for physical experimental education using CV. The force acting on a moving object is visualised as a vector overlapped on the object's trajectory. In [9] and [10], simple setups consisting of a camera and a computer were used to analyse a falling object to determine the acceleration due to gravity g. [10] mentions that the results can be published onto a web page later. In [11], Raspberry Pi 4 with camera module was used to set up an experiment that detects central elastic collisions of two plastic balls, where OpenCV was used to detect collisions. However, all these experiments cannot be considered RTL experiments as they should be performed manually. Also, publishing the results and resetting the experiment are not automated or controlled remotely over the internet.

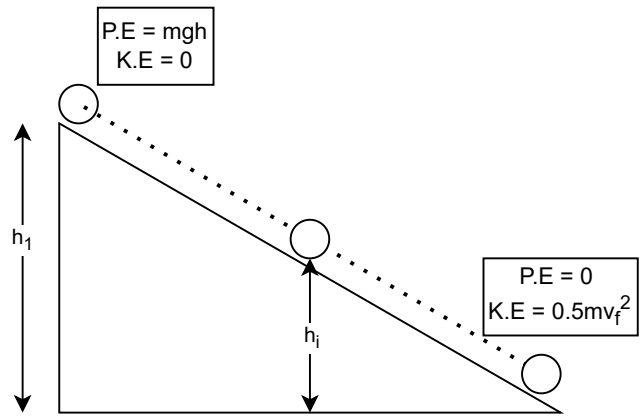
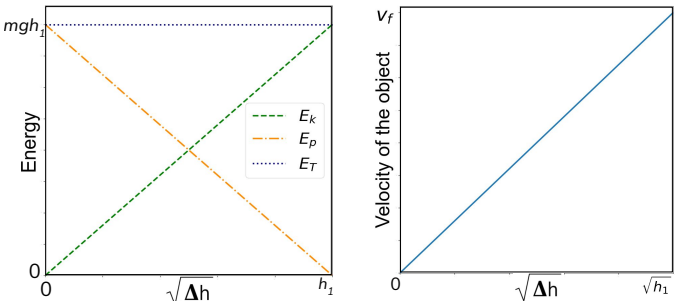


Fig. 1: An example of Conservation of Mechanical Energy



(a) Change in various forms of energy vs $\sqrt{\Delta h}$

(b) Plot of velocity of the object vs $\sqrt{\Delta h}$

Fig. 2: Plots of change in energy and velocity wrt $\sqrt{\Delta h}$

The rest of the paper is structured as follows: Section II describes the physics of the use-case experiment, followed by the description of the hardware designed in Section III. Section IV presents the methodology of the CV-based approach for estimating the object's velocity. Section V shows the IR-based approach for estimating velocity. Results are presented in Section VI while Section VII concludes the paper.

II. Experiment

In this section, the principle of *Conservation of Mechanical Energy* is discussed and later illustrated with an example.

A. Theorem

The principle of conservation of mechanical energy: This states that the mechanical energy of a moving body at any point remains constant throughout its motion.

According to the theorem, under the assumption that there are no resistive forces like friction, the mechanical energy of an object, E_T, which is the sum of its kinetic energy E_k and potential energy E_p, is always a constant, i.e.,

$$E_T = E_p + E_k = \text{constant}. \tag{1}$$

Here the kinetic energy E_k is the energy of motion and is defined as

$$E_k = \frac{1}{2}mv^2, \tag{2}$$

where m is the mass of the point object and v is the instantaneous velocity of the moving object. Also, gravitational potential energy E_p is the energy of the object due to its position relative to the earth and is defined as

$$E_p = mgh, \tag{3}$$

where g is the acceleration due to gravity, and h is the height of the object relative to an arbitrary reference level.

To keep the experiment simple and intuitive, only two forms of energies, gravitational potential energy E_p and translational kinetic energy E_k are considered throughout the experiment. All the discussions and experiments are restricted to point mass objects, i.e., when the object of interest covers a much greater distance than its dimensions, it can be considered a point object.

B. Example

Consider Fig. 1, where a ball is released from rest on a frictionless triangular wedge of vertical height h_1 and fixed to the ground as shown. As the ball moves down the wedge, E_p decreases while E_k increases as the magnitude of the velocity of the ball increases. Consider an intermediate point where the ball is at a height of h_i and has a velocity v_i. Using (1), the total energy at the height h_i is given by

$$E_T = \frac{1}{2}mv_i^2 + mgh_i. \tag{4}$$

On the other hand, the total energy at the initial rest point is entirely because of potential energy so that

$$E_T = mgh_1. \tag{5}$$

Using (4) and (5), we get

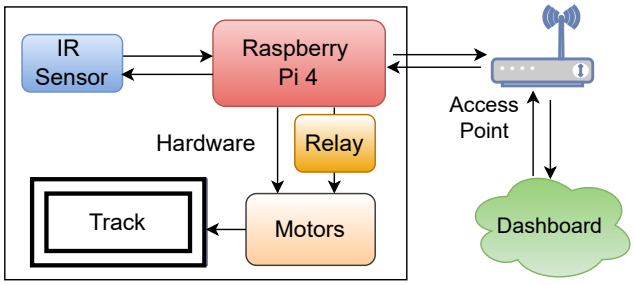
$$\begin{aligned} mgh_1 &= \frac{1}{2}mv_i^2 + mgh_i, \\ \therefore \quad v_i^2 &= \frac{2mg(h_1 - h_i)}{m}, \\ \therefore \quad v_i &= \sqrt{2g\Delta h}, \end{aligned} \tag{6}$$

where $\Delta h = h_1 - h_i$ is the height of the intermediate point wrt topmost point (h_1).

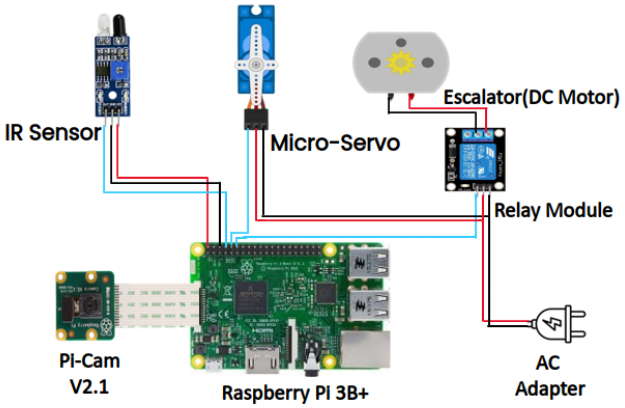
Fig. 2a shows the change in E_T, E_p and E_k as the ball falls down the wedge. It can be observed that the mechanical energy E_T is constant everywhere while E_p and E_k change as the height of the object changes. From (6) and Fig. 2b, it can be observed that $v_i \propto \sqrt{\Delta h}$ indicating that when an object falls down from a starting point (h_1), its velocity increases linearly with square-root of difference of height wrt starting point. Effectively, (1) becomes equivalent to (6) and this will be used to show the conservation of mechanical energy in further sections.

III. Hardware Setup

Figs. 3a and 3b show the block architecture and circuit design of the RTL setup designed to implement the experiment. The setup essentially consists of a 3D modelled track,

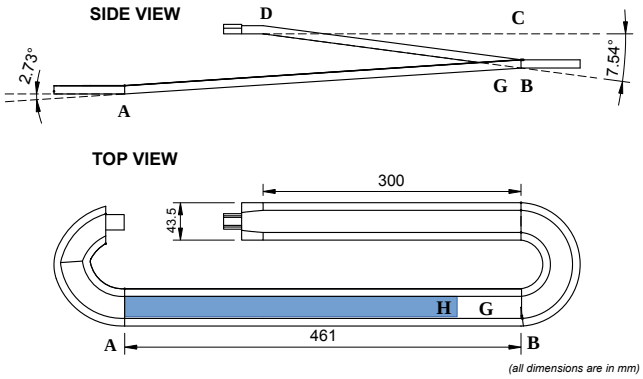


(a) Block architecture of the RTL setup

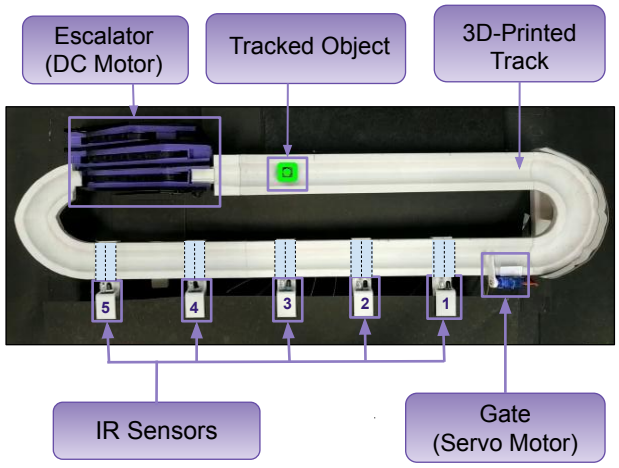


(b) Circuit Diagram of the RTL setup

Fig. 3: Hardware description of the experimental setup



(a) 3D track used for the experiment



(b) Actual Setup

Fig. 4: Setup of the experiment

IR sensors [12], an active high relay module, a micro-servo motor, a DC motor, Raspberry Pi 3B+ [13] and a Raspberry Pi V2.1 camera module [14].

Raspberry Pi 3B+ is the microprocessor used in the RTL setup that interfaces the dashboard with all the sensors and actuators. This microprocessor is connected to a WiFi network to host the experiment online. A 3D track is designed using Fusion 360 [15] and printed with white poly-lactic acid (PLA) material. All the dimensions and coordinates of the points on the track are predetermined as shown in Fig. 4a. The structure consists of a continuous track of two gradually varying slopes along which IR sensors are placed and an escalator. The object, in this case, a stainless steel ball of diameter 18.5 mm and mass 500 g, is set to roll on the track. The escalator, operated by a DC motor, forms a closed path in the track by raising the object to the top position. The servo motor acts as a gate (G), as shown in 4a, to bring the object to a halt and release it from rest later. This gate can be controlled by a user remotely via the dashboard. Point H in Fig. 4a denotes the point where the velocity estimation starts (discussed in the later sections).

The Raspberry Pi camera, a widely-used camera with Raspberry Pi, is used to record at 30 FPS and is fixed vertically above the track. The camera is fixed such that the whole track is captured while recording. Fig. 4b shows the actual setup captured from the camera. All the hardware components are mostly hidden and covered with the background to avoid their visual disturbances during recordings that might affect the velocity estimations. It supports upto 40 FPS with full Field of Vision (FoV) while a maximum of 90 FPS [16] is supported with limited FoV which is undesirable as the whole track can not be captured. So, a mobile camera [17] is used to record the track at higher frame rates of 60, 240 and 480 FPS to analyse the effect of change of FPS on the results. This mobile camera can be replaced with an USB camera that supports higher FPS and can be attached to the experimental setup.

IV. Computer Vision Based Setup

To compute the object's velocity, in our case, the steel ball, a CV-based approach has been used. Using the setup explained in section III, the experiment is recorded while the object is moving on the track as shown in Fig. 4b. CV-based algorithms, including background subtraction, morphological transformation, image filtering, thresholding, and contour detection, are used to track the object. Fig. 5 shows the algorithmic pipeline of the implementation.

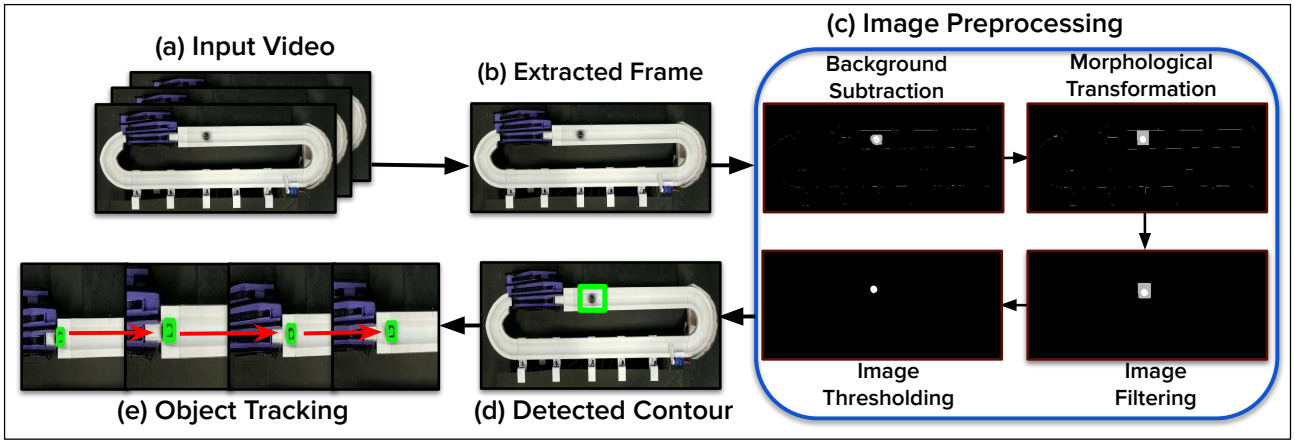


Fig. 5: Algorithmic pipeline of the CV-based setup for object tracking. The arrow in (e) object tracking shows the moving steel ball being tracked in consecutive frames.(Best viewed in color)

A. *Object localization and tracking*

In order to localize and track the object from the video captured, the following steps are taken:

1) Frame Acquisition: The first step is to extract the frames from the video sequentially. The video is recorded from the top view of the whole setup. The extracted frames are stored for further processing.

2) Image Preprocessing: The image preprocessing is a 4-step process which is as follows: i) background subtraction, ii) morphological operation, iii) image filtering, iv) image thresholding.

Our task is to localize the moving steel ball, which is a part of the image foreground. The background is subtracted from the image to get the foreground which mainly contains the steel ball due to its motion. After getting the foreground image, which can still contain noise, morphological operations are used to remove these noises. This method helps to close small holes inside the foreground objects in the frame. The closing morphology method consists of repeated steps of dilation followed by erosion. After this, image filtering processes the edges using a median filter to remove the noise. Finally, the image thresholding process is applied after converting the image into grayscale. As a result, frames are converted into a binary image, and a closed curve is formed around the steel ball. For this process, median thresholding (threshold value set to 127) is used. This process is done for the system to create blobs for the later stage of inspection. Fig. 5(c) shows the output of the frame after the image has been preprocessed.

3) Contour Detection: The preprocessed frame is used to localise the desired object. The closed curves formed around the object are used to determine the location of the steel ball. In any image, *Contours* can be defined simply as a curve joining all the consecutive points along the object's boundary. As in our case, the curves are formed around the steel ball (similar can be seen in Fig. 5d), finding the contours will provide the steel ball's location. As the contour's location is determined in coordinates, a bounding box is drawn to highlight the region of interest, i.e., the localised object and the centroid of the bounding box is stored.

4) Tracking: After determining the object location in one particular frame, the same process mentioned above is applied to all the consecutive frames. In each frame, the object location is determined and tracked further. Fig. 5(e) shows the bounding box (green) around the moving object tracked in each frame.

B. *Calculations*

Algorithm 1 presents our approach to estimate the velocity of the object between 2 frames. Its inputs include the frames captured from the camera, coordinates of the object's centroid for each frame tracked earlier, a conversion factor (d_w) to convert pixel distance to real-life distance (specific for a given track) and the frame rate of the input video (lines 2∼6). Every two consecutive frames are considered and the Euclidean distance between their centroids is calculated (lines 10∼11). Then the velocity of the moving object is estimated (lines 12∼14) using the time taken to travel between two consecutive frames.

V. IR Sensor Based Setup

In the non-CV based setup, sensors are required to measure velocity. In this paper, IR modules are used to measure the velocity of the moving object. HW-201 Infrared Obstacle Avoidance Sensor Module [12] is used for the purpose. Table I shows the specifications of these sensors. IR sensors are placed in rectangular 3D printed cases that are fitted to the track's edges, as shown in Fig. 4b, enabling the sensors to remain in a fixed position. The modules are placed perpendicular to the track, and their effective range is set to 3 cm to ensure accurate detection of the object.

TABLE I: Specifications of HW-201 IR sensor [12]

Power Supply	Distance Range	Sampling Rate	Sensitivity Range	Frequency Range
3-5V DC	2-30 cm	1KHz	800-1100 nm	35-41 KHz

Algorithm 1 Estimate Velocity (V_{est})

procedure ESTVEL(F, C, d_w, FPS)
Inputs:
F: array of captured frames
C: array of x and y coordinates of the centroids (frame-wise)
d_w: conversion factor (pixels to meters)
FPS: frame rate of the recorded video
Output:
V_{est}: Estimated velocities between every two frames
Method:
index = 0
    while index $\leq F - 1$ **do**
        a = Euclidean distance between centroids of index and index+1 frames (in pixels)
        $d_p = d_w \times a$
        $t_p = \frac{1}{\text{FPS}}$
        $V_{est} = \frac{d_p}{t_p}$

A. Calculations

An IR sensor is used to calculate the velocity in a Region of Interest (RoI), where the average velocity is approximated as the object's instantaneous velocity. The RoI is a small rectangular region (<8 cm^2) in front of the IR sensor (marked with dotted lines in blue regions in Fig. 4b), where the sensor can detect the presence of an object. A timer is started when the object passes through an RoI and is incremented till the object leaves the RoI. Let Δt be the total increments of the timer. The estimated velocity of the ball using IR setup (V_{IR}) is

$$V_{IR} = \frac{0.0185}{\Delta t}, \tag{7}$$

where the distance covered by the object will be equal to the diameter of the ball, i.e. 0.0185 m.

VI. Results

In this section, the performance of CV-based implementation is first presented, including the effects of the number of data points considered N and the effects of FPS of the video. Later, the performance comparison between the CV and IR based implementations is presented. For all the experiments in this section, we are using the longest segment of the track, i.e., segment GA of length 420 mm, where the IR sensors are also deployed. Although the ball is released from point G, velocities are only measured from point H which is 50 mm apart from G. This is done to ensure that the object tracker is not disturbed by the movement of the gate G. The segment HA of length 370 mm (shown by the blue region in Fig. 4b) is divided uniformly into N data points and the object's velocities are estimated for these points. Heights h_i are measured with reference to point A (i.e., height of point A = 0). Initial point G is at a height h_1 of 20 mm. The conversion

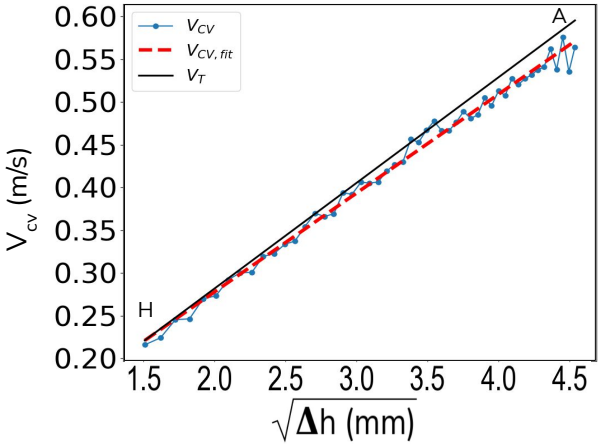


Fig. 6: Plots of $V_{CV}, V_{CV,fit}$ and V_T vs $\sqrt{\Delta h}$ for N = 50 and FPS = 480

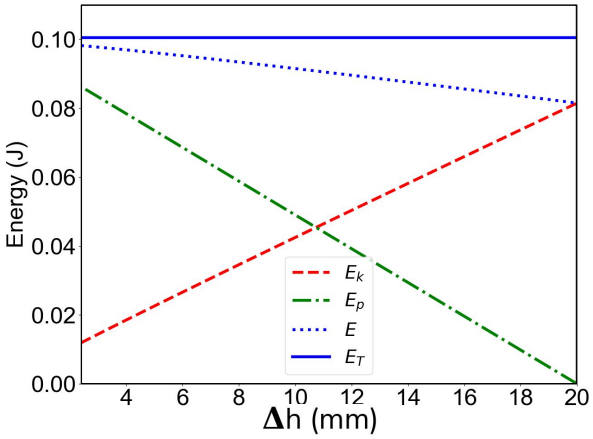
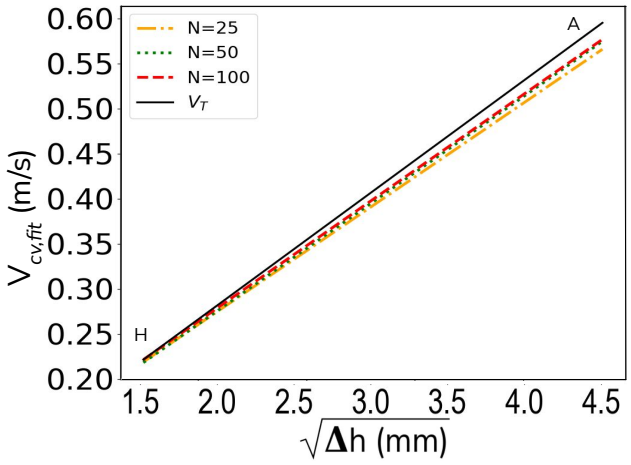


Fig. 7: Plot of E_k, E_p, E_T and E vs Δh for N = 50 and FPS = 480

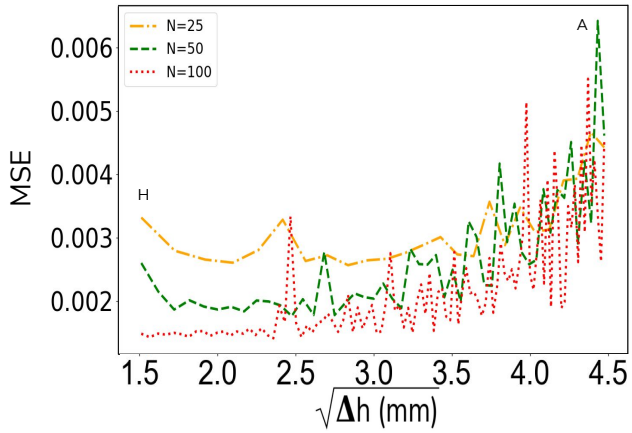
factor d_w is the ratio of length of the segment GA (420 mm) and Euclidean distance of the segment GA (in pixels). In general, students perform an RTL experiment for about 5-10 times in a session. Therefore, the number of experimental runs N_r over which the results are averaged has been set to 10. Unless specified, default values of g, N and frame rate of the video in FPS are 9.8 m/s^2, 50 and 480, respectively. For the CV-based implementation, *OpenCV* [18] is used, which is a well-known open-source library that is extensively used in the domain of CV for image segmentation, object detection and object tracking. It has several built-in functions specifically designed for the purpose used directly in this work.

A. Effects of line fitting

Fig. 6 shows the theoretical velocities V_T and estimated velocities using CV (V_{CV} and $V_{CV,fit}$) at different points on the track as a function of square-root of difference of height of the object wrt point G (i.e., $\sqrt{\Delta h}$). From the figure, it can be observed that the curves are very close and the velocities increase as Δh increases. The second observation is that the



(a) Plot of $V_{CV,fit}$ vs $\sqrt{\Delta h}$ for FPS = 480



(b) Plot of MSE vs $\sqrt{\Delta h}$ for FPS = 480

Fig. 8: Plots for different number of regions (N)

error at various data points is fluctuating. From (6), we have $v_i \propto \sqrt{\Delta h}$. Therefore, to improve the estimated values at individual data points, we used the line of best fit using linear regression $V_{CV,fit}$, which can be seen to have better performance as seen in Fig. 6. It can be observed that the estimated velocities are deviating from V_T as Δh increases. This can be attributed to measurement errors as well as friction along the track.

Fig. 7 shows the corresponding plot of E_k, E_p, E_T and the calculated mechanical energy E vs Δh. Here, E is the sum of E_k and E_p, where E_k is estimated using $V_{CV,fit}$. It can be noted that E_k is non-zero at point H as velocity is not estimated from the release point of the object, i.e, point G. Secondly, E deviates slightly from E_T and this happens due to friction and measurement errors.

B. Effect of the number of data points (N)

Figs. 8a and 8b show the line of best fits and mean square errors (MSE) with respect to $\sqrt{\Delta h}$ for N = 25, 50 and 100. From Fig. 8a, it can be observed that the line of best fit comes closer to the theoretical value as N increases. However, the

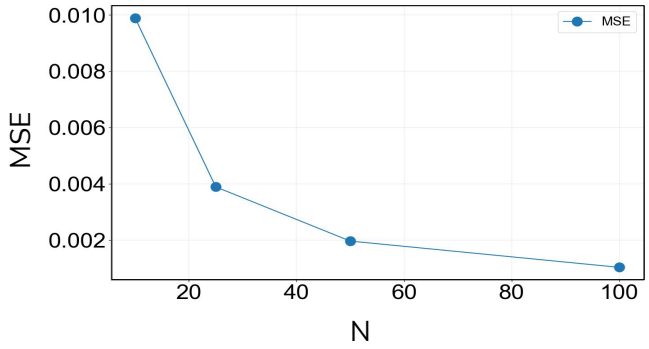


Fig. 9: Plot of MSE vs N for FPS = 480

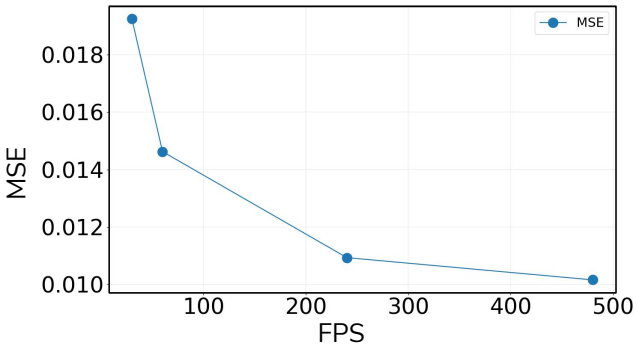


Fig. 10: Plot of MSE vs FPS for N = 10

change from $N = 50$ to $N = 100$ is very small indicating that the line of best fit obtained is nearly optimal and the remaining bias is mainly due to friction. These claims are supported by Fig. 8b, where MSE for $N = 50$ and $N = 100$ are very similar and lower compared to $N = 25$. Similar result can be observed from Fig. 9, which shows the MSE averaged over the segment HA as a function of N.

C. Effect of FPS of the video

Fig. 10 shows the plot of MSE for V_{CV} wrt V_T for N = 10 and FPS = 30, 60, 240 and 480. Low value of N is chosen to gather sufficient frames when FPS = 30. It can be observed that the MSE reduces as the FPS values of the recorded video increase. Secondly, it can be observed that the drop in MSE decreases as FPS increases from 30 to 480 FPS. There is no significant reduction in MSE for videos recorded above 240 FPS.

D. Comparison of CV and IR-based implementations

Table II shows the MSE for velocities estimated using CV (i.e., V_{CV} and $V_{CV,fit}$) and sensor-based (i.e., V_{IR} and $V_{IR,fit}$) implementations wrt V_T where $V_{IR,fit}$ is the line of best fit for velocities estimated using the IR-based implementation. Here, $N_{disc} = 5$ discontinuous regions are chosen for the IR-based implementation that are marked with dotted lines in Fig. 4b. While using the CV-based implementation,

velocities are estimated for $N = 50$ and velocities of all the data points lying in each of the N_{disc} regions are averaged for comparison purpose. It can be observed that the MSE increases with height due to obvious reasons in both cases. Also, line fitting improves the MSE for both implementations. However, best MSE values were obtained for $V_{CV,fit}$ that are almost 10 times better showing that the CV implementation can be used instead of IR-based implementation. Also, we can increase the number of data points as we increase the FPS of the recorded video, allowing us to monitor changes happening at multiple points along the track. However, this is not possible with sensor-based implementations as the number of sensors that can be placed is constrained by the length of the track (a maximum of 12 IR sensors can be placed for this track).

TABLE II: Comparison of velocities obtained using both implementations for N_{disc} = 5, N = 50 and FPS = 480

Region	V_T (m/s)	MSE for V_{IR}	MSE for $V_{IR,fit}$	MSE for V_{CV}	MSE for $V_{CV,fit}$
1	0.25	0.0002	0.0001	0.0004	**0.00005**
2	0.32	0.0008	0.0007	0.0007	**0.00008**
3	0.40	0.0027	0.0013	0.0017	**0.00015**
4	0.48	0.0030	0.0019	0.0022	**0.00023**
5	0.59	0.0034	0.0024	0.0031	**0.00031**

VII. Conclusion

This paper proposed the use of CV for RTL and successfully demonstrated the same for the use-case of *Conservation of Mechanical Energy*. In this experiment, CV is used to estimate the velocities of a moving object on different points along the track, which are are observed to be close to the theoretical velocity values obtained from the principal of mechanical energy conservation. In the proposed CV-based implementation, experiments were captured and processed using OpenCV algorithms. Linear regression was performed on the obtained velocities to improve the MSE wrt the theoretical values. A comparison has been made between the CV and IR sensors-based implementations to compute the velocity of the moving object. After the detailed comparison,the CV-based implementation combined with linear regression outperforms the IR sensors-based setup while computing the velocities. In future, it is intended to study the effects of friction and estimate the coefficient of friction from the estimated velocity plots.

References

[1] "MIT iLabs," http://icampus.mit.edu/projects/ilabs/, Accessed: 28- Mar- 2022.

[2] "RT Labs - NITK," http://rtlabs.nitk.ac.in/?q=page/rt-lab, Accessed: 28- Mar- 2022.

[3] "UNILabs," https://unilabs.dia.uned.es/, Accessed: 28- Mar- 2022.

[4] B. Shankar et al., "Remote triggered virtual laboratory for Hooke's law using LabVIEW," in *39th Annual Conference of the IEEE Industrial Electronics Society*, 2013, pp. 3729–3734.

[5] Restivo et al., "A Remote Laboratory in Engineering Measurement," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 12, pp. 4836–4843, 2009.

[6] L de la Torre et al., "Two web-based laboratories of the FisL@bs network: Hooke's and Snell's laws," *European Journal of Physics*, vol. 32, no. 2, pp. 571–584, feb 2011.

[7] Sousa et al., "An Integrated Reusable Remote Laboratory to Complement Electronics Teaching," *IEEE Transactions on Learning Technologies*, vol. 3, no. 3, pp. 265–271, 2010.

[8] G. Nakagawa et al., "A Learning Material for Physics Experiment with High-Accuracy Using Computer Vision Technique," in *5th Intl Conf on Applied Computing and Information Technology (ACIT)*, 2017, pp. 86–89.

[9] K. Pattanashetty et al., "Web-based physics experiments in dynamics using image processing," in *IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*, 2016, pp. 1–5.

[10] G. Illeperuma et al., "Computer vision based object tracking as a teaching aid for high school physics experiments," in *4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, 2017, pp. 1–6.

[11] M. Hajba et al, "Elastic collisions visualization using OpenCV object motion tracking," in *43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*, 2020, pp. 1573–1578.

[12] "HW-201 IR sensor," https://www.inro-electronics.com/infrared-object-detection-sensor, Accessed: 30- Mar- 2022.

[13] "Raspberry Pi 3 Model B+," https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/, Accessed: 30- Mar- 2022.

[14] "Raspberry Pi Camera Module 2," https://www.raspberrypi.com/products/camera-module-v2/, Accessed: 30- Mar- 2022.

[15] "Fusion 360," https://www.autodesk.com/products/fusion-360/overview?term=1-YEAR&tab=subscription, Accessed: 30- Mar- 2022.

[16] "Pi Camera specifications," https://picamera.readthedocs.io/en/release-1.13/fov.html, Accessed: 31- Mar- 2022.

[17] "OnePlus 8T," https://www.oneplus.in/8t, Accessed: 28- Mar- 2022.

[18] "OpenCV: Python library for real-time computer vision," https://opencv.org/, Accessed: 26 Mar, 2022.